

[SREES] SeminarSKI - SystemModel Dokumentacija

Generated by Doxygen 1.9.4



<b>1 Namespace Index</b>	<b>1</b>
1.1 Namespace List	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Namespace Documentation</b>	<b>7</b>
4.1 SystemModel Namespace Reference	7
4.1.1 Typedef Documentation	7
4.1.1.1 AdmittanceMatrix	8
4.1.1.2 dfidx	8
4.1.1.3 fi	8
4.1.2 Enumeration Type Documentation	8
4.1.2.1 ThreePhaseLoadConfigurationsType	8
4.1.2.2 TypeOfBus	8
4.1.3 Function Documentation	9
4.1.3.1 operator<<()	9
<b>5 Class Documentation</b>	<b>11</b>
5.1 SystemModel::Bus Class Reference	11
5.1.1 Constructor & Destructor Documentation	12
5.1.1.1 Bus()	12
5.1.2 Member Function Documentation	12
5.1.2.1 getActivePower()	12
5.1.2.2 getReactivePower()	12
5.1.2.3 getTypeOfBus()	13
5.1.2.4 getVoltageMagnitude()	13
5.1.2.5 getVoltagePhase()	13
5.1.2.6 setActivePower()	13
5.1.2.7 setReactivePower()	13
5.1.2.8 setVoltageMagnitude()	15
5.1.2.9 setVoltagePhase()	15
5.2 SystemModel::SystemModel Class Reference	15
5.2.1 Constructor & Destructor Documentation	17
5.2.1.1 SystemModel()	17
5.2.2 Member Function Documentation	17
5.2.2.1 addBus()	17
5.2.2.2 addCapacitorBank()	17
5.2.2.3 addGenerator()	18
5.2.2.4 addLine()	18
5.2.2.5 addLoad()	18

---

5.2.2.6 addSlackGenerator()	19
5.2.2.7 addTransformer()	19
5.2.2.8 getAdmittanceMatrix()	20
5.2.2.9 getBus()	20
5.2.2.10 getBusFunctions()	20
5.2.2.11 getDerivativesOfBusFunctions()	20
5.2.2.12 getNumberOfBuses()	21
5.2.2.13 hasSlackBeenAssigned()	21
5.2.3 Friends And Related Function Documentation	21
5.2.3.1 operator<<	21
<b>6 File Documentation</b>	<b>23</b>
6.1 main.cpp File Reference	23
6.1.1 Function Documentation	23
6.1.1.1 main()	23
6.2 systemModel.cpp File Reference	24
6.2.1 Macro Definition Documentation	24
6.2.1.1 PI	24
6.3 systemModel.h File Reference	24
6.4 systemModel.h	26
<b>Index</b>	<b>29</b>

# Chapter 1

## Namespace Index

### 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">SystemModel</a> . . . . .	7
---------------------------------------	---



## Chapter 2

# Class Index

### 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">SystemModel::Bus</a>	11
<a href="#">SystemModel::SystemModel</a>	15





## Chapter 3

# File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

<a href="#">main.cpp</a>	23
<a href="#">systemModel.cpp</a>	24
<a href="#">systemModel.h</a>	24



## Chapter 4

# Namespace Documentation

### 4.1 SystemModel Namespace Reference

#### Classes

- class [Bus](#)
- class [SystemModel](#)

#### Typedefs

- using [fi](#) = std::pair< std::function< double(std::vector< double >)>, std::function< double(std::vector< double >)>>> >
- using [dfidx](#) = std::pair< std::vector< std::function< double(std::vector< double >)>> >, std::vector< std::function< double(std::vector< double >)>> >>
- using [AdmittanceMatrix](#) = std::vector< std::tuple< uint8\_t, uint8\_t, std::complex< double > > >

#### Enumerations

- enum class [TypeOfBus](#) { [Slack](#) , [PV](#) , [PQ](#) }
- enum class [ThreePhaseLoadConfigurationsType](#) { [Star](#) , [GroundedStar](#) , [Delta](#) }

#### Functions

- std::ostream & [operator<<](#) (std::ostream &stream, const [SystemModel](#) &systemModel)  
*Output stream operator overload*

#### 4.1.1 Typedef Documentation

#### 4.1.1.1 AdmittanceMatrix

```
using SystemModel::AdmittanceMatrix = typedef std::vector<std::tuple<uint8_t, uint8_t, std::complex<double>>>
```

#### 4.1.1.2 dfidx

```
using SystemModel::dfidx = typedef std::pair<std::vector<std::function<double(std::vector<double>)>>>
```

#### 4.1.1.3 fi

```
using SystemModel::fi = typedef std::pair<std::function<double(std::vector<double>)>>, std::function<double(std::vector<double>)>>
```

### 4.1.2 Enumeration Type Documentation

#### 4.1.2.1 ThreePhaseLoadConfigurationsType

```
enum class SystemModel::ThreePhaseLoadConfigurationsType [strong]
```

##### Enumerator

Star	
GroundedStar	
Delta	

#### 4.1.2.2 TypeOfBus

```
enum class SystemModel::TypeOfBus [strong]
```

##### Enumerator

Slack	
PV	
PQ	

### 4.1.3 Function Documentation

#### 4.1.3.1 operator<<()

```
std::ostream & SystemModel::operator<< (
    std::ostream & stream,
    const SystemModel & systemModel )
```

Output stream operator overload

##### Parameters

<i>stream</i>	Output stream object
<i>systemModel</i>	<a href="#">SystemModel</a> object to be printed to the stream

##### Returns



## Chapter 5

# Class Documentation

### 5.1 SystemModel::Bus Class Reference

```
#include <systemModel.h>
```

Collaboration diagram for SystemModel::Bus:

SystemModel::Bus
<div><div>+ Bus()</div><div>+ getTypeOfBus()</div><div>+ setVoltageMagnitude()</div><div>+ setVoltagePhase()</div><div>+ setActivePower()</div><div>+ setReactivePower()</div><div>+ getVoltageMagnitude()</div><div>+ getVoltagePhase()</div><div>+ getActivePower()</div><div>+ getReactivePower()</div></div>

#### Public Member Functions

- [Bus](#) ([TypeOfBus](#) typeOfBus)
- [TypeOfBus](#) [getTypeOfBus](#) () const
- void [setVoltageMagnitude](#) (double voltageMagnitude)  
*Sets the value at which the voltage amplitude for the given bus should be maintained.*
- void [setVoltagePhase](#) (double voltagePhase)  
*Sets the value at which the voltage phase for the given bus should be maintained.*
- void [setActivePower](#) (double activePower)

- Sets the value at which the active power for the given bus should be maintained.*

  - void [setReactivePower](#) (double reactivePower)
- Sets the value at which the rective power for the given bus should be maintained.*

  - std::optional< double > [getVoltageMagnitude](#) () const
- Gets the value at which the voltage magnitude for the given bus should be maintained.*

  - std::optional< double > [getVoltagePhase](#) () const
- Gets the value at which the voltage phase for the given bus should be maintained.*

  - std::optional< double > [getActivePower](#) () const
- Gets the value at which the active power for the given bus should be maintained.*

  - std::optional< double > [getReactivePower](#) () const
- Gets the value at which the rective power for the given bus should be maintained.*

## 5.1.1 Constructor & Destructor Documentation

### 5.1.1.1 Bus()

```
SystemModel::Bus::Bus (
    TypeOfBus typeOfBus ) [inline]
```

## 5.1.2 Member Function Documentation

### 5.1.2.1 getActivePower()

```
std::optional< double > SystemModel::Bus::getActivePower ( ) const
```

Gets the value at which the active power for the given bus should be maintained.

#### Returns

Value of active power for the bus

### 5.1.2.2 getReactivePower()

```
std::optional< double > SystemModel::Bus::getReactivePower ( ) const
```

Gets the value at which the rective power for the given bus should be maintained.

#### Returns

Value of reactive power for the bus



### 5.1.2.3 getTypeOfBus()

```
TypeOfBus SystemModel::Bus::getTypeOfBus ( ) const [inline]
```

### 5.1.2.4 getVoltageMagnitude()

```
std::optional< double > SystemModel::Bus::getVoltageMagnitude ( ) const
```

Gets the value at which the voltage magnitude for the given bus should be maintained.

#### Returns

Value of voltage magnitude of the bus

### 5.1.2.5 getVoltagePhase()

```
std::optional< double > SystemModel::Bus::getVoltagePhase ( ) const
```

Gets the value at which the voltage phase for the given bus should be maintained.

#### Returns

Value of voltage phase of the bus

### 5.1.2.6 setActivePower()

```
void SystemModel::Bus::setActivePower (
    double activePower )
```

Sets the value at which the active power for the given bus should be maintained.

#### Parameters

<i>activePower</i>	Value of active power for the bus
--------------------	-----------------------------------

### 5.1.2.7 setReactivePower()

```
void SystemModel::Bus::setReactivePower (
    double reactivePower )
```

Sets the value at which the rective power for the given bus should be maintained.

## Parameters

<i>reactivePower</i>	Value of reactive power for the bus
----------------------	-------------------------------------

**5.1.2.8 setVoltageMagnitude()**

```
void SystemModel::Bus::setVoltageMagnitude (
    double voltageMagnitude )
```

Sets the value at which the voltage amplitude for the given bus should be maintained.

## Parameters

<i>voltageMagnitude</i>	Value of voltage magnitude of the bus
-------------------------	---------------------------------------

**5.1.2.9 setVoltagePhase()**

```
void SystemModel::Bus::setVoltagePhase (
    double voltagePhase )
```

Sets the value at which the voltage phase for the given bus should be maintained.

## Parameters

<i>voltagePhase</i>	Value of voltage phase of the bus
---------------------	-----------------------------------

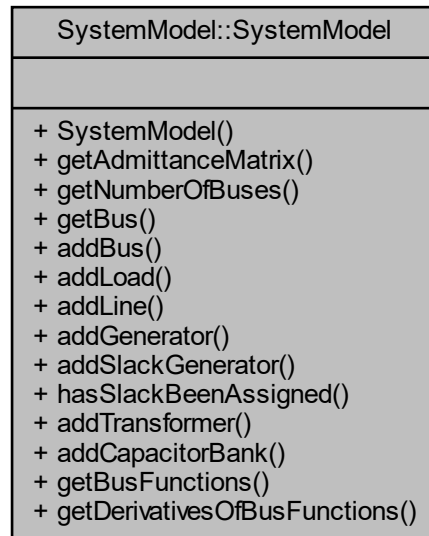
The documentation for this class was generated from the following files:

- [systemModel.h](#)
- [systemModel.cpp](#)

**5.2 SystemModel::SystemModel Class Reference**

```
#include <systemModel.h>
```

Collaboration diagram for SystemModel::SystemModel:



## Public Member Functions

- [SystemModel](#) (uint8\_t maxNumberOfBuses)
- [AdmittanceMatrix](#) [getAdmittanceMatrix](#) () const
- uint8\_t [getNumberOfBuses](#) () const
- [Bus](#) & [getBus](#) (uint8\_t busNumber)  
*Gets the bus with the given bus number*
- void [addBus](#) ([TypeOfBus](#) typeOfBus)  
*Adds a bus to the system*
- void [addLoad](#) (uint8\_t busNumber, double activePower, double reactivePower)  
*Adds a load to a bus*
- void [addLine](#) (uint8\_t busNumber1, uint8\_t busNumber2, double r, double x, double b)  
*Adds a line between buses*
- void [addGenerator](#) (uint8\_t busNumber, double voltageMagnitude, double activePower)  
*Adds a generator to a bus*
- void [addSlackGenerator](#) (uint8\_t busNumber, double voltageMagnitude, double voltagePhase)  
*Adds a generator to the slack bus*
- bool [hasSlackBeenAssigned](#) () const  
*Check whether the slack bus has been assigned*
- void [addTransformer](#) (uint8\_t busNumber1, uint8\_t busNumber2, double r, double x, double g, double b, double n)  
*Adds a transformer between buses*
- void [addCapacitorBank](#) (uint8\_t busNumber, double c, [ThreePhaseLoadConfigurationsType](#) configuration↔  
Type)  
*Adds a capacitor bank to a bus*
- [fi](#) [getBusFunctions](#) (uint8\_t busNumber) const

*Gets the bus functions ( $f_{i\_P}$  and  $f_{i\_Q}$ ) for the desired bus*

- [dfidx getDerivativesOfBusFunctions](#) (uint8\_t busNumber) const

*Gets the derivates of the bus functions ( $df_{i\_P}/dx$  and  $df_{i\_Q}/dx$ ) for the desired bus (two rows of the Jacobian associated with the given bus functions)*

## Friends

- std::ostream & [operator<<](#) (std::ostream &stream, const [SystemModel](#) &systemModel)

## 5.2.1 Constructor & Destructor Documentation

### 5.2.1.1 SystemModel()

```
SystemModel::SystemModel::SystemModel (
    uint8_t maxNumberOfBuses ) [inline]
```

## 5.2.2 Member Function Documentation

### 5.2.2.1 addBus()

```
void SystemModel::SystemModel::addBus (
    TypeOfBus typeOfBus )
```

Adds a bus to the system

#### Parameters

<i>typeOfBus</i>	Type of the bus (Slack, PV, PQ) to be added to the system
------------------	---

### 5.2.2.2 addCapacitorBank()

```
void SystemModel::SystemModel::addCapacitorBank (
    uint8_t busNumber,
    double c,
    ThreePhaseLoadConfigurationsType configurationType )
```

Adds a capacitor bank to a bus

## Parameters

<i>busNumber</i>	Ordinal number of the desired bus
<i>c</i>	One phase capacitance of the bank
<i>configurationType</i>	Three phase load configuration type (delta, star, grounded star) of the bank

**5.2.2.3 addGenerator()**

```
void SystemModel::SystemModel::addGenerator (
    uint8_t busNumber,
    double voltageMagnitude,
    double activePower )
```

Adds a generator to a bus

## Parameters

<i>busNumber</i>	Ordinal number of the desired bus
<i>voltageMagnitude</i>	Voltage magnitude on which the given bus should be maintained

<param name="activePower"Active power on which the given bus should be maintained>

**5.2.2.4 addLine()**

```
void SystemModel::SystemModel::addLine (
    uint8_t busNumber1,
    uint8_t busNumber2,
    double r,
    double x,
    double b )
```

Adds a line between buses

## Parameters

<i>busNumber1</i>	Ordinal number of the first bus
<i>busNumber2</i>	Ordinal number of the second bus
<i>r</i>	Series resistance of the transmission line PI equivalent
<i>x</i>	Series reactance of the transmission line PI equivalent
<i>b</i>	Shunt susceptance of the transmission line PI equivalent

**5.2.2.5 addLoad()**

```
void SystemModel::SystemModel::addLoad (
    uint8_t busNumber,
```

```
double activePower,
double reactivePower )
```

Adds a load to a bus

#### Parameters

<i>busNumber</i>	Ordinal number of the desired bus
<i>activePower</i>	Active power drawn by the load
<i>reactivePower</i>	Reactive power drawn by the load

#### 5.2.2.6 addSlackGenerator()

```
void SystemModel::SystemModel::addSlackGenerator (
    uint8_t busNumber,
    double voltageMagnitude,
    double voltagePhase )
```

Adds a generator to the slack bus

#### Parameters

<i>busNumber</i>	Ordinal number of the desired bus
<i>voltageMagnitude</i>	Voltage magnitude on which the given bus should be maintained
<i>voltagePhase</i>	Voltage phase on which the given bus should be maintained

#### 5.2.2.7 addTransformer()

```
void SystemModel::SystemModel::addTransformer (
    uint8_t busNumber1,
    uint8_t busNumber2,
    double r,
    double x,
    double g,
    double b,
    double n )
```

Adds a transformer between buses

#### Parameters

<i>busNumber1</i>	Ordinal number of the first bus
<i>busNumber2</i>	Ordinal number of the second bus
<i>r</i>	Series resistance of the transformer PI equivalent
<i>x</i>	Series reactance of the transformer PI equivalent
<i>g</i>	Shunt conductance of the transformer PI equivalent
<i>b</i>	Shunt susceptance of the transformer PI equivalent
<i>n</i>	Transformer turns ratio

### 5.2.2.8 getAdmittanceMatrix()

```
AdmittanceMatrix SystemModel::SystemModel::getAdmittanceMatrix ( ) const [inline]
```

### 5.2.2.9 getBus()

```
SystemModel::Bus & SystemModel::SystemModel::getBus (
    uint8_t busNumber )
```

Gets the bus with the given bus number

#### Parameters

<i>busNumber</i>	Ordinal number of the desired bus
------------------	-----------------------------------

#### Returns

[Bus](#) with the given bus number

### 5.2.2.10 getBusFunctions()

```
SystemModel::fi SystemModel::SystemModel::getBusFunctions (
    uint8_t busNumber ) const
```

Gets the bus functions (fi\_P and fi\_Q) for the desired bus

#### Parameters

<i>busNumber</i>	Ordinal number of the desired bus
------------------	-----------------------------------

#### Returns

[Bus](#) functions for the given bus in the form of std::pair of functions, where both functions have a std::vector of doubles as parameters and return a double

### 5.2.2.11 getDerivativesOfBusFunctions()

```
SystemModel::dfidx SystemModel::SystemModel::getDerivativesOfBusFunctions (
    uint8_t busNumber ) const
```



Gets the derivates of the bus functions (dfi\_P/dx and dfi\_Q/dx) for the desired bus (two rows of the Jacobian associated with the given bus functions)

#### Parameters

<i>busNumber</i>	Ordinal number of the desired bus
------------------	-----------------------------------

#### Returns

Derivates of the bus functions for the given bus in the form of std::pair of std::vector-s of functions, where both functions have a std::vector of doubles as parameters and return a double

#### 5.2.2.12 getNumberOfBuses()

```
uint8_t SystemModel::SystemModel::getNumberOfBuses ( ) const [inline]
```

#### 5.2.2.13 hasSlackBeenAssigned()

```
bool SystemModel::SystemModel::hasSlackBeenAssigned ( ) const
```

Check whether the slack bus has been assigned

#### Returns

True if the slack bus has been assigned and false otherwise

### 5.2.3 Friends And Related Function Documentation

#### 5.2.3.1 operator<<

```
std::ostream & operator<< (
    std::ostream & stream,
    const SystemModel & systemModel ) [friend]
```

The documentation for this class was generated from the following files:

- [systemModel.h](#)
- [systemModel.cpp](#)

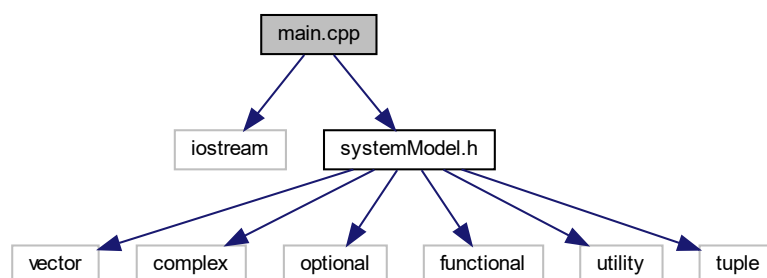


## Chapter 6

# File Documentation

### 6.1 main.cpp File Reference

```
#include <iostream>
#include "systemModel.h"
Include dependency graph for main.cpp:
```



### Functions

- int [main](#) ()

#### 6.1.1 Function Documentation

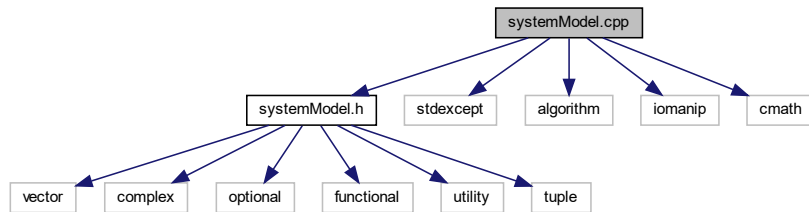
##### 6.1.1.1 main()

```
int main ( )
```

## 6.2 systemModel.cpp File Reference

```
#include "systemModel.h"
#include <stdexcept>
#include <algorithm>
#include <iomanip>
#include <cmath>
```

Include dependency graph for systemModel.cpp:



### Macros

- #define **PI** 4 \* std::atan(1.0)

### 6.2.1 Macro Definition Documentation

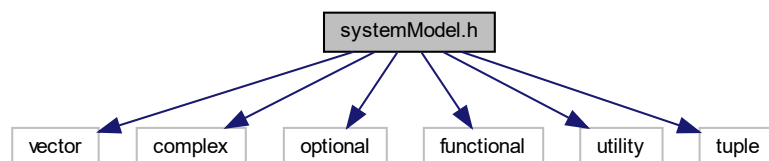
#### 6.2.1.1 PI

```
#define PI 4 * std::atan(1.0)
```

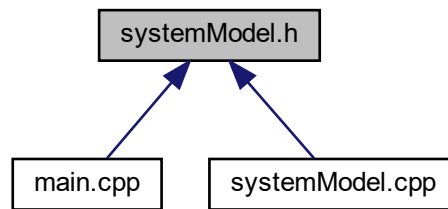
## 6.3 systemModel.h File Reference

```
#include <vector>
#include <complex>
#include <optional>
#include <functional>
#include <utility>
#include <tuple>
```

Include dependency graph for systemModel.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [SystemModel::Bus](#)
- class [SystemModel::SystemModel](#)

## Namespaces

- namespace [SystemModel](#)

## Typedefs

- using [SystemModel::fi](#) = std::pair< std::function< double(std::vector< double >)>, std::function< double(std::vector< double >)>> >
- using [SystemModel::dfidx](#) = std::pair< std::vector< std::function< double(std::vector< double >)>>, std::vector< std::function< double(std::vector< double >)>> >
- using [SystemModel::AdmittanceMatrix](#) = std::vector< std::tuple< uint8\_t, uint8\_t, std::complex< double > > >

## Enumerations

- enum class [SystemModel::TypeOfBus](#) { [SystemModel::Slack](#) , [SystemModel::PV](#) , [SystemModel::PQ](#) }
- enum class [SystemModel::ThreePhaseLoadConfigurationsType](#) { [SystemModel::Star](#) , [SystemModel::GroundedStar](#) , [SystemModel::Delta](#) }

## Functions

- std::ostream & [SystemModel::operator<<](#) (std::ostream &stream, const SystemModel &systemModel)  
*Output stream operator overload*

## 6.4 systemModel.h

[Go to the documentation of this file.](#)

```

1 #pragma once
2 #include <vector>
3 #include <complex>
4 #include <optional>
5 #include <functional>
6 #include <utility>
7 #include <tuple>
8
9
10
11 namespace SystemModel {
12     using fi = std::pair<std::function<double(std::vector<double>)>,
13         std::function<double(std::vector<double>)>>;
14
15
16     using dfidx = std::pair<std::vector<std::function<double(std::vector<double>)>>,
17         std::vector<std::function<double(std::vector<double>)>>>;
18
19
20     using AdmittanceMatrix = std::vector<std::tuple<uint8_t, uint8_t, std::complex<double>>>;
21
22
23
24     enum class TypeOfBus { Slack, PV, PQ };
25
26
27
28     enum class ThreePhaseLoadConfigurationsType { Star, GroundedStar, Delta };
29
30
31
32     class Bus {
33     public:
34         TypeOfBus typeOfBus;
35         std::optional<double> voltageMagnitude;
36         std::optional<double> voltagePhase;
37         std::optional<double> activePower;
38         std::optional<double> reactivePower;
39
40         Bus(TypeOfBus typeOfBus) : typeOfBus{ typeOfBus } {}
41
42         TypeOfBus getTypeOfBus() const {
43             return typeOfBus;
44         }
45
46         void setVoltageMagnitude(double voltageMagnitude);
47
48         void setVoltagePhase(double voltagePhase);
49
50         void setActivePower(double activePower);
51
52         void setReactivePower(double reactivePower);
53
54         std::optional<double> getVoltageMagnitude() const;
55
56         std::optional<double> getVoltagePhase() const;
57
58         std::optional<double> getActivePower() const;
59
60         std::optional<double> getReactivePower() const;
61     };
62
63
64     class SystemModel {
65     public:
66         AdmittanceMatrix admittanceMatrix;
67         uint8_t numberOfBuses{};
68         std::vector<Bus> buses;
69         const uint8_t maxNumberOfBuses;
70         bool checkForConnectionBetweenToBuses(uint8_t busNumber1, uint8_t busNumber2) const;
71
72         SystemModel(uint8_t maxNumberOfBuses) : maxNumberOfBuses{ maxNumberOfBuses } {}
73
74         AdmittanceMatrix getAdmittanceMatrix() const {
75             return admittanceMatrix;
76         }
77
78         uint8_t getNumberOfBuses() const {
79             return numberOfBuses;
80         }
81     };

```

```
81         Bus& getBus(uint8_t busNumber);
82
83     void addBus(.TypeOfBus typeOfBus);
84
85     void addLoad(uint8_t busNumber, double activePower, double reactivePower);
86
87     void addLine(uint8_t busNumber1, uint8_t busNumber2, double r, double x, double b);
88
89     friend std::ostream& operator <<(std::ostream& stream, const SystemModel& systemModel);
90
91     void addGenerator(uint8_t busNumber, double voltageMagnitude, double activePower);
92
93     void addSlackGenerator(uint8_t busNumber, double voltageMagnitude, double voltagePhase);
94
95     bool hasSlackBeenAssigned() const;
96
97     void addTransformer(uint8_t busNumber1, uint8_t busNumber2, double r, double x, double g,
98 double b, double n);
99
100     void addCapacitorBank(uint8_t busNumber, double c, ThreePhaseLoadConfigurationsType
101 configurationType);
102
103     fi getBusFunctions(uint8_t busNumber) const;
104
105     dfidx getDerivativesOfBusFunctions(uint8_t busNumber) const;
106
107 };
108
109 std::ostream& operator <<(std::ostream& stream, const SystemModel& systemModel);
110 }
```





# Index

- addBus
  - SystemModel::SystemModel, 17
- addCapacitorBank
  - SystemModel::SystemModel, 17
- addGenerator
  - SystemModel::SystemModel, 18
- addLine
  - SystemModel::SystemModel, 18
- addLoad
  - SystemModel::SystemModel, 18
- addSlackGenerator
  - SystemModel::SystemModel, 19
- addTransformer
  - SystemModel::SystemModel, 19
- AdmittanceMatrix
  - SystemModel, 7
- Bus
  - SystemModel::Bus, 12
- Delta
  - SystemModel, 8
- dfidx
  - SystemModel, 8
- fi
  - SystemModel, 8
- getActivePower
  - SystemModel::Bus, 12
- getAdmittanceMatrix
  - SystemModel::SystemModel, 20
- getBus
  - SystemModel::SystemModel, 20
- getBusFunctions
  - SystemModel::SystemModel, 20
- getDerivativesOfBusFunctions
  - SystemModel::SystemModel, 20
- getNumberOfBuses
  - SystemModel::SystemModel, 21
- getReactivePower
  - SystemModel::Bus, 12
- getTypeOfBus
  - SystemModel::Bus, 12
- getVoltageMagnitude
  - SystemModel::Bus, 13
- getVoltagePhase
  - SystemModel::Bus, 13
- GroundedStar
  - SystemModel, 8
- hasSlackBeenAssigned
  - SystemModel::SystemModel, 21
- main
  - main.cpp, 23
- main.cpp, 23
  - main, 23
- operator<<
  - SystemModel, 9
  - SystemModel::SystemModel, 21
- PI
  - systemModel.cpp, 24
- PQ
  - SystemModel, 8
- PV
  - SystemModel, 8
- setActivePower
  - SystemModel::Bus, 13
- setReactivePower
  - SystemModel::Bus, 13
- setVoltageMagnitude
  - SystemModel::Bus, 15
- setVoltagePhase
  - SystemModel::Bus, 15
- Slack
  - SystemModel, 8
- Star
  - SystemModel, 8
- SystemModel, 7
  - AdmittanceMatrix, 7
  - Delta, 8
  - dfidx, 8
  - fi, 8
  - GroundedStar, 8
  - operator<<, 9
  - PQ, 8
  - PV, 8
  - Slack, 8
  - Star, 8
  - SystemModel::SystemModel, 17
  - ThreePhaseLoadConfigurationsType, 8
  - TypeOfBus, 8
- systemModel.cpp, 24
  - PI, 24
- systemModel.h, 24
  - SystemModel::Bus, 11
  - Bus, 12

- getActivePower, [12](#)
- getReactivePower, [12](#)
- getTypeOfBus, [12](#)
- getVoltageMagnitude, [13](#)
- getVoltagePhase, [13](#)
- setActivePower, [13](#)
- setReactivePower, [13](#)
- setVoltageMagnitude, [15](#)
- setVoltagePhase, [15](#)
- SystemModel::SystemModel, [15](#)
  - addBus, [17](#)
  - addCapacitorBank, [17](#)
  - addGenerator, [18](#)
  - addLine, [18](#)
  - addLoad, [18](#)
  - addSlackGenerator, [19](#)
  - addTransformer, [19](#)
  - getAdmittanceMatrix, [20](#)
  - getBus, [20](#)
  - getBusFunctions, [20](#)
  - getDerivativesOfBusFunctions, [20](#)
  - getNumberOfBuses, [21](#)
  - hasSlackBeenAssigned, [21](#)
  - operator<<, [21](#)
  - SystemModel, [17](#)
- ThreePhaseLoadConfigurationsType
  - SystemModel, [8](#)
- TypeOfBus
  - SystemModel, [8](#)